# BONUS ASSIGNMENT QUESTIONS

[ASM'20]

BONUS Assignment Questions (Mainstream and Bioinformatics)

Faculty of Computer & Information Sciences
Due date: 28 December 2020 Midnight
Course: Assembly language CSW-353
Instructor: Dr. Karim Emara – Dr. Salsabil Amin

## General instructions:

Regarding your file:

1) Submit **only running** code that you have tested before.
2) Use the provided template to write your code. Do not use any other templates.
3) Submit **only** the content of **main.asm** file. You have to press "Submit" button in the form.
4) Submit your code through this Google form link
5) Do **not** use for ALL questions:
    a. Conditional directives, such as .if, .while, etc …
6) Do not hard-code answers. It is clear that it will be considered unacceptable behavior.

**This assignment is intended for *individual* contribution. Sharing ideas or part of answers is considered plagiarism and will not be tolerated (= ZERO mark for the entire assignment). All submissions will be checked for plagiarism automatically.**

Marks will be **deducted** for not following any of the above submission rules (-5/instruction).

**PLEASE WATCH THE AUTOGRADER EXPLANATION VIDEO CAREFULLY BEFORE SUBMITTING YOUR CODE**

**Notes:**

1. All the sample runs were taken from the assignment's template.
2. Array values are not read in one line. They are read line-by-line (i.e., separated by new lines).

Good luck
Assembly team 2020

BONUS Assignment Questions (Mainstream and Bioinformatics)

Faculty of Computer & Information Sciences
Due date: 28 December 2020 Midnight
Course: Assembly language CSW-353
Instructor: Dr. Karim Emara – Dr. Salsabil Amin

**Q1.** In the procedure called “Q1”, write assembly code that reads a number *n* from the user and draw a diamond shape using that number. Its shape is defined by the following:

- The central line of the diamond contains 2n-1 asterisks.
- The difference between each row and the adjacent row is 2 asterisks.
- Spaces are only on the left of the diamond, and the difference between each row and the adjacent row is 1 space.
- Notice that there is one space before the central line.

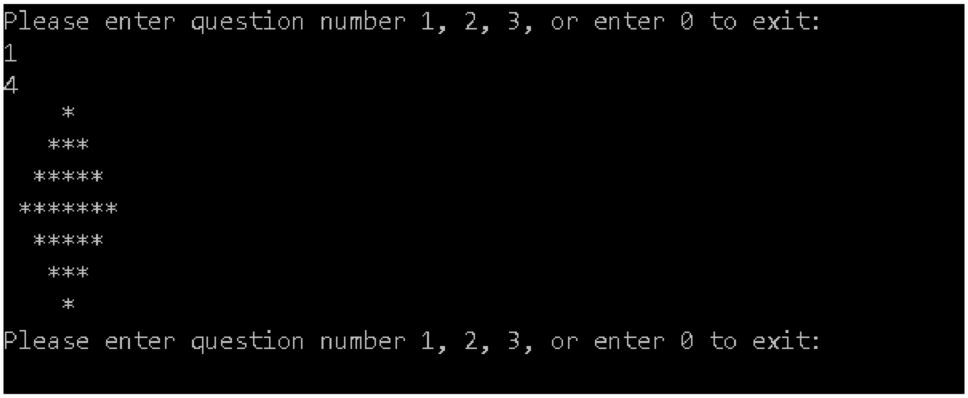
**Sample Input:**

4

**Sample Output:**

```
    *
   ***
  *****
 *******
  *****
   ***
    *
```

**Sample Run:**



```
Please enter question number 1, 2, 3, or enter 0 to exit:
1
4
    *
   ***
  *****
 *******
  *****
   ***
    *
Please enter question number 1, 2, 3, or enter 0 to exit:
```

*Figure 1 question-1 sample run*

Good luck
Assembly team 2020

**Q2.** In the procedure called "Q2", write assembly code that takes a number from user & reverses the numerals in the integer.

**Constraints:**

- You must read an integer from the user using CALL READINT or CALL READDEC, but not CALL READSTRING.
- You can use MUL and DIV.

**Sample Input:**

Enter a number: 7623

**Sample Output:**

Reverse of the number: 3267

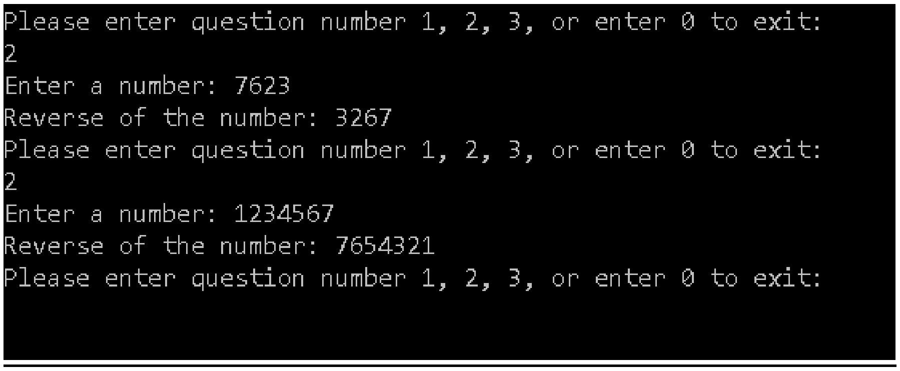
**Sample Input:**

Enter a number: 1234567

**Sample Output:**

Reverse of the number: 7654321

**Sample Run:**



```
Please enter question number 1, 2, 3, or enter 0 to exit:
2
Enter a number: 7623
Reverse of the number: 3267
Please enter question number 1, 2, 3, or enter 0 to exit:
2
Enter a number: 1234567
Reverse of the number: 7654321
Please enter question number 1, 2, 3, or enter 0 to exit:
```

*Figure 2 question-2 sample run*

**Q3.** In the procedure called "Q3", write assembly code that calculates the sum of the powers of the digits of a given **INTEGER.**

Given a number, decompose it to its constituting digits then calculate $n^3$ where n is a digit in the given number. Finally, sum the powers.

**Examples:**

56 → $(5^3 + 6^3)$

327 → $(3^3 + 2^3 + 7^3\ )$

123456789 → $(1^3 + 2^3 + 3^3 + \cdots)$

**Constraints:**

- You must read an integer from the user using CALL READINT or CALL READDEC, but not CALL READSTRING.
- You can use MUL and DIV.

**Sample Input:**

Enter A NUMBER: 56

**Sample Output:**

THE RESULT = : 341

**Sample Input:**

Enter A NUMBER: 327

**Sample Output:**

THE RESULT = : 378

Good luck
Assembly team 2020

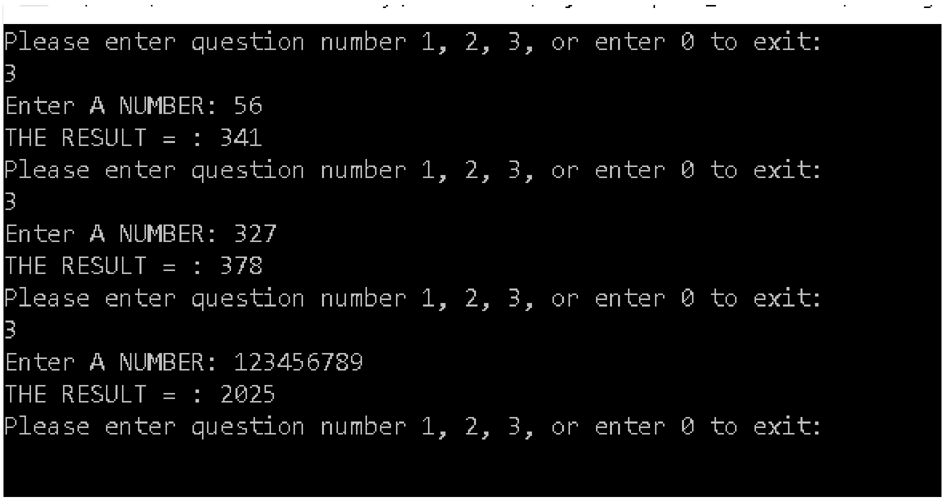
**Sample Input:**

Enter A NUMBER: 123456789

**Sample Output:**

THE RESULT = : 2025

**Sample Run:**



```
Please enter question number 1, 2, 3, or enter 0 to exit:
3
Enter A NUMBER: 56
THE RESULT = : 341
Please enter question number 1, 2, 3, or enter 0 to exit:
3
Enter A NUMBER: 327
THE RESULT = : 378
Please enter question number 1, 2, 3, or enter 0 to exit:
3
Enter A NUMBER: 123456789
THE RESULT = : 2025
Please enter question number 1, 2, 3, or enter 0 to exit:
```

*Figure 3 question-3 sample run*

Hints:

- WriteInt is an Irvine function that prints an integer value that must be stored in EAX register (number's sign is printed).
- ReadInt is an Irvine function that reads an integer from the keyboard and stores it in EAX register (the input integer is signed).
- WriteDec is an Irvine function that prints an integer value that must be stored in EAX register (number's sign is not printed).
- ReadDec is an Irvine function that reads an integer from the keyboard and stores it in EAX register (the input integer is not signed).
- WriteChar is an Irvine function that prints a character that must be stored in AL register.
- ReadString is an Irvine function that reads a string from the keyboard, stopping when the user presses the Enter key. Pass the offset of a buffer in EDX and set ECX to the maximum

number of characters the user can enter. The procedure returns the count of the number of characters typed by the user in EAX.

- WriteString is an Irvine function that writes a string to the console. Pass the offset of a buffer in EDX.
- ReadHex used to read a hexadecimal value from the user. The value after the read is stored in EAX register.
- WriteHex used to write a hexadecimal value to the screen. The value to be displayed is stored in EAX register before calling this procedure.
- ReadChar is used to read a char from the console. The value read from the console is placed in "al" register.
- More about these functions and similar ones can be found in section 5.3 of the book.

Good luck
Assembly team 2020